

I. General Remarks Concerning This Response

Claims 1-40 are currently pending in the present application. Claims 3, 21, and 39 are objected to as being dependent upon a rejected base claim but would be allowable if rewritten in independent form. No claims have been amended, added, or canceled herein. Reconsideration of the claims is requested.

II. Summary of Present Invention

The present invention is directed to management of a distributed data processing system. A network management framework adapts management database operations so as to minimize the impact on system performance that is caused by the system management operations. In particular, performance adjustments occur dynamically in accordance with the current needs of the system management infrastructure. A database is emulated in memory, and applications that consume data from the database are presented with an interface with reduced overhead for accessing the data. In order to expedite a database access, some objects are pre-fetched into a cache.

A skeletonization mechanism is presented in which some objects are merely represented in the cache as a skeleton object. A skeleton object holds only a fraction of an object's full complement of data; the skeleton object remains associated with its complete object, also termed its corresponding full object or non-skeleton object, which remains stored within its original location within a database.

When it is determined that a network management framework component may need access to objects within a database, some full objects may be pre-fetched into a cache while other full objects may be merely represented within the cache by their corresponding skeleton objects. Hence, some of the full objects that could have been pre-fetched into the cache are represented within the

cache only by their corresponding skeleton objects. A cache comprising both full objects and skeleton objects may be termed a skeleton cache or a skeletonized cache.

5 In a given implementation of a network management framework, the types of objects that are used by the network management framework components will dictate the types of corresponding skeleton objects. In order for the network management framework to be as flexible as possible, each type of skeleton object is structured in accordance with a skeleton definition associated with the skeleton object; the skeleton definition allows a skeleton handler, i.e. a manager component for the operations on skeleton objects, to determine which attributes within an object should be represented within its corresponding skeleton object. The attributes of an object that are not fully valued within the skeleton object are termed skeleton attributes.

15 In addition, again so that the network management framework can be as flexible as possible, the present invention employs skeleton policies. A skeleton policy defines when a full object should be skeletonized, i.e. when an object should be retrieved from persistent storage into memory, e.g., a cache, as a skeleton object rather than as a copy of the full object. Various network management framework components refer to the skeleton policy such that the skeleton policy directs the operation of the network management framework components with respect to a particular object, i.e. the network management framework components determine when to skeletonize an object based on information within the skeleton policy.

III. 35 U.S.C. § 102(b)-Anticipation-Nori et al.

The Office action has rejected claims 1, 2, 4-9, 17-20, 22-27, 35-38, and 40 under 35 U.S.C. § 102(b) as anticipated by Nori et al., "Apparatus and method for storage of object collections in a database system", U.S. Patent No. 6,061,690, filed 10/31/1977, issued 05/09/2000. This rejection is respectfully traversed.

The rejection of independent claim 1 states in its entirety:

As to claims 1, 19, and 37, Nori discloses a method, apparatus with means, and program product for creating a skeleton cache that stores full and skeleton objects, where a skeleton object has at least one attribute that is dataless, and has a corresponding full object without the skeleton cache. See Fig. 4, and col. 10 line 15 to col. 11 line 21, in particular col. 10 lines 55-58 and col. 10 line 65 to col. 11 line 2.

The rejection does not fully recite all of the features that are recited within independent claim 1. For ease of reference, independent claim 1 recites in its entirety:

1. A method for managing objects in a data processing system, the method comprising:
creating a skeleton cache; and
storing a first object in the skeleton cache, wherein a skeleton cache stores skeleton objects and/or full objects, wherein a full object is an object in which each attribute within the full object has a data value, wherein a skeleton object is an object in which at least one attribute within the skeleton object is dataless, and wherein a skeleton object has a corresponding full object that is stored without the skeleton cache but within the data processing system.

As is apparent from the copy of the rejection that is included hereinabove, the rejection does not provide any detail or any argument as to which elements of the system that is disclosed in Nori et al. correspond to the claimed elements of the present application. The rejection points to entire sections of Nori et al. rather than comparing specific features of Nori et al. against specific claim elements. The rejection points to an

entire column of text within Nori et al. for all of the claim elements, although seven lines of text are particularly referenced. Thus, the rejection does not provide any guidance on the manner in which one is to interpret Nori et al. as
5 anticipating the claimed invention. Applicant asserts that the lack of explanatory detail in the rejection with respect to individual claim elements mirrors the lack of disclosure in Nori et al. with respect to some of the claimed features.

Applicant asserts that Nori et al. completely fails to
10 disclose at least one feature of claim 1, notwithstanding the reference from the rejection into the text in Nori et al. as supposedly anticipating all claimed features. Applicant asserts that Nori et al. merely discloses a typical cache but does not disclose a skeleton cache as claimed in the present patent
15 application, e.g., as recited by the first element of claim 1. Assuming *arguendo* that Nori et al. discloses something more than a typical cache, then Nori et al. does not disclose a skeleton object as claimed in the present patent application.

Specifically, Nori et al. does not disclose the feature of
20 "wherein a skeleton object is an object in which at least one attribute within the skeleton object is dataless", as recited within claim 1. In order to emphasize the fact that Nori et al. does not disclose a skeleton object as recited within claim 1, Applicant includes hereinbelow a copy of the portion of Nori et al.
25 al. that was applied against claim 1. For ease of reference, the portion of Nori et al. that is mentioned in the rejection of claim 1 reads in its entirety (emphasis added):

FIG. 4 is a flow diagram of a method of retrieving an object that contains a collection attribute from a storage location. In an embodiment, FIG. 4 represents processing steps carried out when an object having a collection attribute is retrieved from non-volatile storage, such as disk, to volatile storage, such as an object cache of a database server that needs the object, or local memory of a machine or process that needs the object.
30
35

In block 402, an application program requests an information object. Block 402 is intended to broadly reflect any process of requesting an information object that is under management by a database server or other process or program that implements the invention. For example, block 402 represents a request by a database server to obtain an information object that is needed to carry out an application program running under control of the server.

In block 404, the object's ADT [Abstract Data Type] is examined to determine whether the object has a collection attribute. In an embodiment, this step is carried out by retrieving metadata that describes the type of the object from a table of the database system. For example, each information object includes a type identifier that is stored in association with data values and methods of the object. Using the type identifier, the database system looks up metadata about that type of object in an object type table. The database system retrieves a type descriptor object corresponding to the object's type from that table, and interprets the information in the type descriptor object. The type descriptor object has fields that indicate the data type of each attribute of an object. Collections including nested tables and varrays are indicated by unique codes in the attribute fields of the type descriptor object.

If the metadata indicates that the object does not have any collection attributes, then as shown in block 406, the entire object is retrieved from its location in non-volatile storage and copied into volatile storage, such as the object cache. At this point, the object is ready for further use by an application program, and the method is considered complete, as shown in block 408.

If the object contains a collection attribute, it is desirable not to retrieve the values or other contents of the collection attribute until they are actually needed by the database server or a related application program. Collection attributes, such as nested tables, may be quite large. Accordingly, if the entire collection attribute is always retrieved when an object is requested, it is possible that the available volatile storage space associated with the requesting program or process will be insufficient. The address space may overflow.

Therefore, as shown in block 410, when a requested object has a collection attribute, only an identifier of the collection attribute is retrieved and returned to the requesting program or process. The identifier is a handle or reference to the values of the collection. For example, in an embodiment, the identifier is a value comprising a base table identifier concatenated with a "setid", and when

the requested object is delivered to the calling program, the object contains the identifier in the position of the collection attribute.

Block 412 represents processing steps in which the contents of the collection attribute remain untouched until they are needed by a calling program or process. Thus, the collection attribute remains in non-volatile storage until an element of the collection is actually requested or needed by the database server or a related program. In that event, as shown in block 414, in one embodiment, a partition of the non-volatile storage device that contains the needed element value is retrieved into local volatile storage, or into the object cache. Preferably, the partition is pinned or locked so that other programs or processes cannot modify the partition while the collection element value is in use. In block 416, the value of the element is delivered to the database server or the calling program.

Nori et al. discloses that a collection attribute is a large amount of data, such as a large variable array or a nested table, and that it is desirable to delay the retrieval of the collection attribute, e.g., from non-volatile memory to an object cache of a database server, until actually requested or needed by a program.

In particular, Nori et al. states:

... when a requested object has a collection attribute, only an identifier of the collection attribute is retrieved and returned to the requesting program or process. The identifier is a handle or reference to the values of the collection. ... when the requested object is delivered to the calling program, the object contains the identifier in the position of the collection attribute.

In other words, when an object is going to be retrieved into a cache, and the object contains a collection attribute, then the retrieved object contains a handle or a pointer to the contents of the collection attribute rather than the collection attribute itself. Although there is nothing explicitly stated in the rejection, it appears as if the rejection is implying that the retrieved object with the delayed retrieval of the collection attribute's contents is equivalent to the claimed feature of the present invention in which a skeleton object has at least one attribute that is dataless.

Applicant disagrees with the implied argument. Independent claim 1 specifically recites that at least one attribute of the skeleton object is dataless; if the claim is interpreted using the ordinary meaning of the terms, then the claim must be interpreted as stating that at least one attribute of the skeleton object has no data. Referring to the specification of the present patent application for the meaning of the term "dataless", "dataless" is explained by stating that "a skeleton attribute is dataless, i.e. does not contain or has not been given or assigned a data value".

Thus, the system that is disclosed by the present invention is not equivalent to the system that is disclosed in Nori et al.. Nori et al. discloses that a collection attribute is replaced with an identifier, such as a handle or a reference pointer, that indicates the location of the values of the collection, e.g., in non-volatile memory. This is significant because the system in Nori et al. uses the identifier to retrieve the collection attribute's contents; the identifier indicates that the collection attribute's contents is not currently stored together with the remainder of the cached object.

In contrast, the present invention does not use a reference pointer or any data value in the skeleton attribute. This is significant because the system of the present invention uses information outside of the skeleton object, e.g., a skeleton definition, to indicate when or how to retrieve the data that is represented by the skeleton object, i.e. the full object.

Moreover, the system that is disclosed in Nori et al. always replaces the collection attribute with an identifier; Nori et al. does not disclose otherwise. In addition, the system that is disclosed in Nori et al. only delays the retrieval of a collection attribute; this operation is not performed on other data types.

In contrast, the present invention allows for a skeleton attribute to be used for any abstract data type, thereby providing for delayed retrieval of any type of data. In addition, large tables or variable arrays are not necessarily forced to be skeleton attributes. Thus, the present invention is more flexible and robust. Although it could be argued that Nori et al. and the present system have a common goal of delaying data retrieval until necessary, they present two different solutions. However, since Nori et al. does not disclose the claimed features of the present invention, Nori et al. cannot anticipate the claimed invention.

With respect to dependent claims 2, 4-9, 17, and 18, Nori et al. does not disclose, at a minimum, the subject matter in independent claim 1 from which these dependent claims depend. Thus, Nori et al. also fails to disclose the features of the dependent claims because these dependent claims include the features of independent claim 1.

Moreover, the dependent claims recite additional elements, and these elements also fail to be disclosed in Nori et al.. Applicant notes that the rejections of dependent claims are also as non-specific as the rejection of independent claim 1 from which they depend. In fact, the rejections of the dependent claims completely fail to indicate any particular locations in Nori et al. at which the features of the dependent claims are found. Again, the lack of detail in the rejections mirrors the lack of disclosure in Nori et al..

With respect to dependent claim 2, the rejection states that the claimed feature is disclosed "since the ADT [abstract data type] identifies if an object has a collection attribute, which is dataless when requested to the cache." As argued above by Applicant, the collection attribute is not dataless when the object is retrieved within the system that is disclosed by Nori et al.; the collection attribute contains an identifier, which is

similar to a handle or a pointer reference. Hence, the collection attribute is not dataless. More importantly, claim 2 recites:

5 retrieving a skeleton definition associated with the first object, wherein a skeleton definition is associated with an first object's type, wherein a skeleton definition indicates whether an attribute within the first object is a skeleton attribute, and wherein a skeleton attribute is a dataless attribute.

10 Nori et al. does not disclose a skeleton definition. In the present invention, a skeleton definition is a data item that is not identical to a skeleton object nor a full object, and it is not stored within a skeleton object or a full object. A skeleton
15 definition is associated with an object's type and "indicates whether an attribute within the first object is a skeleton attribute". In the system that is disclosed in Nori et al., there is no need for a skeleton definition because all collection attributes are replaced with an identifier for subsequent
20 retrieval; it is hardwired into the system because the disclosed operation is simply based on an attribute being a collection attribute. The present system is more robust because a skeleton attribute indicates whether an attribute within a cached object should be skeletonized.

25 With respect to dependent claim 4, the rejection states that the claimed feature is disclosed "since any number of objects may be requested, and for any cache access it is determined if the data is within the object in the cache." The rejection appears to argue that claim 4 is merely reciting a typical cache
30 operation of determining if an object is stored within the cache or within some other datastore. However, claim 4 recites more than this; claim 4 recites "... and determining whether the second object resides in the skeleton cache as a full object or as a skeleton object". First, Nori et al. does not disclose
35 skeleton objects as is claimed in the present invention. Second, Nori et al. does not make any type of determination of whether an

object within a cache is present in the cache as a full object or as a skeleton object.

With respect to dependent claim 5, the rejection states that the claimed feature is disclosed "since clearly if the attribute is hit in the cache it is retrieved from there." The argument in the rejection of claim 5, which depends from claim 4, is based on an argument similar to the argument in the rejection of claim 4. Applicant notes that Nori et al. does not disclose the claimed feature in claim 5 because Nori et al. does not disclose any step "in response to a determination that the second object resides in the skeleton cache as a full object", as recited in claim 5.

With respect to dependent claims 6-8, the rejection states that the claimed features are disclosed "since the cited sections describe retrieving attributes from the cache or from the secondary storage if not in the cache." The argument in the rejection of claims 6-8, which depend from claim 4, is based on an argument similar to the argument in the rejection of claim 4. Applicant notes that Nori et al. does not disclose the claimed features because Nori et al. does not disclose any step "in response to a determination that the second object resides in the skeleton cache as a skeleton object", as recited in claim 6, nor any step "in response to a determination that the second attribute is a skeleton attribute", as recited in claims 7 and 8.

With respect to dependent claim 9, the rejection states that the claimed feature is disclosed "since a skeleton cache is created when an object has a collection attribute". The argument in the rejection is a complete misrepresentation of the disclosure of Nori et al., which discloses that a cache exists and then implements the operations concerning a collection attribute. Nori et al. does not disclose anything similar to a skeleton policy nor a step of "determining whether to create a skeleton cache in accordance with a skeleton policy, wherein the skeleton policy comprises one or more configurable conditions for

determining whether to create a skeleton cache", as recited in claim 9.

With respect to dependent claim 18, the rejection states that the claimed feature is disclosed "since a skeleton cache to the extent recited is formed when an object has a collection attribute". Again, the argument in the rejection is a complete misrepresentation of the disclosure of Nori et al., which discloses that a cache exists and then implements the operations concerning a collection attribute. Nori et al. does not disclose anything similar to a skeleton policy nor a step of "configuring a skeleton policy for the database access interface component, wherein the skeleton policy comprises one or more configurable conditions for determining whether to create a skeleton cache", as recited in claim 18.

Claims 1-18 are directed to a method; claims 19-36 are directed to an apparatus; and claims 37-40 are directed to a computer program product. The Office action uses an anticipation argument against claims 19, 20, 22-27, 35-38, and 40 by relying the argument that is used against claims 1, 2, 4-9, 17, and 18. Applicant's arguments with respect to the rejection of claims 1, 2, 4-9, 17, and 18 are similarly applicable against the rejection of claims 19, 20, 22-27, 35-38, and 40.

Nori et al. clearly does not disclose features as required by the language of the claims of the present application. As stated at MPEP § 2131: "A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). "The identical invention must be shown in as complete detail as is contained in the ... claim." *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989). Hence, Nori et al. cannot be used as an anticipatory reference, and the

rejection of claims 1, 2, 4-9, 17-20, 22-27, 35-38, and 40 has been overcome, whereby Applicant requests the withdrawal of the rejection.

5 **IV. 35 U.S.C. § 103(a)—Obviousness—Nori et al.**

 The Office action has rejected claims 10-12 and 28-30 under 35 U.S.C. § 103(a) as unpatentable over Nori et al.. This rejection is traversed.

 The rejection states in its entirety:

10 As to claims 10-12 and 28-30, Nori does not disclose
configurable conditions based on identity of a user or their
class, or of a device. However, these are well known
15 identifications used for determining accessibility to a
system, and as such would have been included in conditions
required for determining whether to create a skeleton cache
to the extent recited. Thus it would have been obvious to
one of ordinary skill in the art at the time of the
invention to base the determination on these conditions,
20 because they were known to be used for restricting or
providing accessibility to a system.

While it may have been well known in the prior art to have configurable conditions based on an identity of a user or of a device, the fact that "these are well known identifications used
25 for determining accessibility to a system" is irrelevant with respect to the claimed features, so it is unclear why this statement is included within the rejection. Claims 10-12 depend from claim 9; the claimed features in claims 10-12 refer to the step in claim 9 in which a determination is made "whether to
30 create a skeleton cache in accordance with a skeleton policy". The skeleton policy is concerned with whether or not to create a skeleton cache, not whether certain users or devices should be able to access certain resources in the system.

More importantly, the rejection does not provide any citation to any prior art for the assertion that "as such would have been included in conditions required for determining whether to create a skeleton cache". Since there is no citation of any prior art to support this assertion, it must be assumed that the association of the claimed configurable conditions with the creation of a skeleton cache was obtained from Applicant's own disclosure, and as such, the rejection has relied upon an improper amount of hindsight in applying Applicant's own disclosure against Applicant's own claims.

Furthermore, the rejection does not provide any argument as to how the system that is disclosed in Nori et al. could be modified to include the claimed features, e.g., by particularly pointing out specific features in the system of Nori et al. that could be hypothetically modified to include the claimed features. Assuming *arguendo* that the system of Nori et al. could be modified to include the claimed features, Applicant asserts that doing so would completely change the principal of operation of the disclosed system. Nori et al. discloses that its novel features with the collection attributes can be implemented whenever the objects with the collection attributes are copied into volatile memory from non-volatile memory, possibly as an object cache for a database server. If the claimed feature of "determining whether to create a skeleton cache" were implemented in the hypothetical system as argued by the rejection, then the system would operate such that it made a decision of whether or not to copy objects into volatile memory. However, this is illogical because the system as disclosed in Nori et al. would not be able to operate under such conditions.

Claims 1-18 are directed to a method; claims 19-36 are directed to an apparatus; and claims 37-40 are directed to a computer program product. The Office action primarily uses an obviousness argument against claims 28-30 by relying the argument

that is used against claims 10-12. Applicant's arguments with respect to the rejection of claims 10-12 are similarly applicable against the rejection of claims 28-30.

5 Examiner bears the burden of establishing a *prima facie* case of obviousness

The examiner bears the burden of establishing a *prima facie* case of obviousness based on the prior art when rejecting claims under 35 U.S.C. § 103. *In re Fritch*, 972 F.2d 1260, 23

10 U.S.P.Q.2d 1780 (Fed. Cir. 1992). Only when a *prima facie* case of obviousness is established does the burden shift to the applicant to produce evidence of nonobviousness. *In re Oetiker*, 977 F.2d 1443, 1445, 24 U.S.P.Q.2d 1443, 1444 (Fed. Cir. 1992); *In re Rijckaert*, 9 F.3d 1531, 1532, 28 U.S.P.Q.2d 1955, 1956 (Fed. Cir. 15 1993). If the Patent Office does not produce a *prima facie* case of unpatentability, then without more the applicant is entitled to grant of a patent. *In re Oetiker*, 977 F.2d 1443, 1445, 24 U.S.P.Q.2d 1443, 1444 (Fed. Cir. 1992); *In re Grabiak*, 769 F.2d 729, 733, 226 U.S.P.Q. 870, 873 (Fed. Cir. 1985). In response to 20 an assertion of obviousness by the Patent Office, the applicant may attack the Patent Office's *prima facie* determination as improperly made out, present objective evidence tending to support a conclusion of nonobviousness, or both. *In re Fritch*, 972 F.2d 1260, 1265, 23 U.S.P.Q.2d 1780, 1783 (Fed. Cir. 1992).

25 Nori et al. clearly fails to show or to suggest a feature of the present invention as currently claimed, notwithstanding the assertion by the Office action, thereby rendering Nori et al. incapable of being used as a primary reference as argued by the current rejection. As should be recognized, because Nori et al. 30 fails to disclose or to suggest the claimed features against which Nori et al. was applied, the rejection fails to fulfill the requirements of a proper obviousness argument.

With respect to claims 10-12 and 28-30, Applicant respectfully submits that the teachings of the applied reference cannot be modified to produce the claimed invention. Hence, a rejection of the claims cannot be based upon the cited prior art to establish a *prima facie* case of obviousness. Therefore, a rejection of the claims under 35 U.S.C. § 103(a) has been shown to be insupportable in view of the cited prior art, and the claims are patentable over the applied reference. Applicant respectfully requests the withdrawal of the rejection of the claims.

V. 35 U.S.C. § 103(a)—Obviousness—Nori in view of Veres

The Office action has rejected claims 13-16 and 31-34 under 35 U.S.C. § 103(a) as unpatentable over Nori et al. in view of Veres et al., "Method and system for managing data in computer memory", U.S. Patent Number 6,609,186 B1, filed 04/06/1999, issued 08/19/2003. This rejection is traversed.

The rejection states in its entirety:

As to claims 13-16 and 31-34, Nori does not disclose configurable conditions based on available memory, network bandwidth, nor temporal evaluation of updates. However, Veres describes an analogous system in that it also provides for reducing a stored object to some subset of its original size (see Abstract, Fig. 5). Veres teaches that parts of an object are advantageously deleted as a function of constraints in memory available (col. 1 lines 38-43), network bandwidth (col. 2 lines 34-38), and temporal evaluation of updates (col. 2 lines 22-26). An artisan would have recognized that these could provide the same functionality to the system of Nori because it reduces the size of a stored object. Thus it would have been obvious to one of ordinary skill in the art at the time of the invention to base determinations on these conditions, because they were known to benefit the controlling of reducing object size in a storage.

Claims 13-16 depend from claim 9; the claimed features in claims 13-16 refer to the step in claim 9 in which a determination is made "whether to create a skeleton cache in accordance with a skeleton policy". The skeleton policy is concerned with whether or not to create a skeleton cache and not with processes for controlling which items should be stored in the cache.

Furthermore, the rejection does not provide any argument as to how the system that is disclosed in Nori et al. could be modified to include the features of Veres et al., e.g., by particularly pointing out specific features in the system of Nori et al. that could be hypothetically modified to include the features. Assuming *arguendo* that the system of Nori et al. could be modified to include the features, Applicant asserts that doing so would completely change the principal of operation of the disclosed system. Nori et al. discloses that its novel features with the collection attributes can be implemented whenever the objects with the collection attributes are copied into volatile memory from non-volatile memory, possibly as an object cache for a database server. If the claimed feature of "determining whether to create a skeleton cache" were implemented in the hypothetical system as argued by the rejection, then the system would operate such that it made a decision of whether or not to copy objects into volatile memory. However, this is illogical because the system as disclosed in Nori et al. would not be able to operate under such conditions.

Claims 1-18 are directed to a method; claims 19-36 are directed to an apparatus; and claims 37-40 are directed to a computer program product. The Office action primarily uses an obviousness argument against claims 31-34 by relying the argument that is used against claims 13-16. Applicant's arguments with respect to the rejection of claims 13-16 are similarly applicable against the rejection of claims 31-34.

With respect to claims 13-16 and 31-34, Applicant respectfully submits that the teachings of the applied references cannot be modified to produce the claimed invention. Hence, a rejection of the claims cannot be based upon the cited prior art to establish a *prima facie* case of obviousness. Therefore, a rejection of the claims under 35 U.S.C. § 103(a) has been shown to be insupportable in view of the cited prior art, and the claims are patentable over the applied references. Applicant respectfully requests the withdrawal of the rejection of the claims.

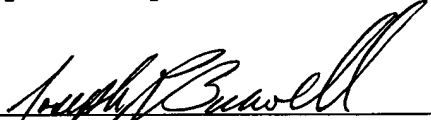
VI. Conclusion

It is respectfully urged that the present patent application is patentable, and Applicant kindly requests a Notice of Allowance.

For any other outstanding matters or issues, the examiner is urged to call or fax the below-listed telephone numbers to expedite the prosecution and examination of this application.

DATE: September 3, 2004

Respectfully submitted,


Joseph R. Burwell
Reg. No. 44,468
ATTORNEY FOR APPLICANT

Law Office of Joseph R. Burwell
P.O. Box 28022
Austin, Texas 78755-8022
Voice: 866-728-3688 (866-PATENT8)
Fax: 866-728-3680 (866-PATENT0)
Email: joe@burwell.biz